Automatic Question-Answer Generation for Long-Tail Knowledge

Rohan Kumar*, Youngmin Kim*, Sunitha Ravi*,
Haitian Sun, Christos Faloutsos, Ruslan Salakhutdinov, Minji Yoon†
Carnegie Mellon University, Pittsburgh, PA, USA
{youngmik,rohankum,selvansr,haitians,christos,rsalakhu,minjiy}@andrew.cmu.edu

ABSTRACT

Pretrained Large Language Models (LLMs) have gained significant attention for addressing open-domain Question Answering (QA). While they exhibit high accuracy in answering questions related to common knowledge, LLMs encounter difficulties in learning about uncommon long-tail knowledge (tail entities). Since manually constructing QA datasets demands substantial human resources, the types of existing QA datasets are limited, leaving us with a scarcity of datasets to study the performance of LLMs on tail entities. In this paper, we propose an automatic approach to generate specialized QA datasets for tail entities and present the associated research challenges. We conduct extensive experiments by employing pretrained LLMs on our newly generated long-tail QA datasets, comparing their performance with and without external resources including Wikipedia and Wikidata knowledge graphs.

KEYWORDS

Question and Answering, Large Language Models, Long-Tail Knowledge, Knowledge Graphs

ACM Reference Format:
Rohan Kumar*, Youngmin Kim*, Sunitha Ravi*,, Haitian Sun, Christos Faloutsos, Ruslan Salakhutdinov, Minji Yoon† . 2023. Automatic Question-Answer Generation for Long-Tail Knowledge. In *KnowledgeNLP-KDD'23: KDD Workshop on Knowledge Augmented Methods for NLP, August 07, 2023, Long Beach, CA.* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Open-domain Question Answering (QA) [6, 10], which involves answering questions regarding common knowledge, has long been a challenging task in the fields of natural language understanding, information retrieval, and related domains [13, 22]. Large language models (LLMs) trained on internet text effectively capture a wide range of world knowledge, encompassing both widely known facts and domain-specific information. These models have achieved remarkable success in QA tasks, eliminating the need for external documents during inference by implicitly storing knowledge in their parameters [7, 14, 18].

However, the impressive achievements of LLMs in QA tasks are primarily observed with regard to common concepts that frequently appear on the internet (referred to as "head entities"), which are thus more likely to be learned effectively by LLMs during pretraining time. Conversely, when it comes to dealing with long-tail knowledge, which encompasses rarely occurring entities (referred to as "tail entities"), LLMs struggle to provide accurate answers and often exhibit hallucination issues [5]. Due to the predominant focus of most QA datasets on head entities [3, 6, 10], research investigating the performance of LLMs on long-tail knowledge has been limited. Recently, Kandpal et al. [7] conducted a study to bridge this gap by constructing dedicated QA datasets for tail entities. Their approach involved leveraging the entity frequency in Wikipedia to define tail entities and quantitatively demonstrating the limitations of LLMs in handling such entities.

Wikipedia documents [12] and Wikidata knowledge graphs [23] are the primary external resources from which QA models acquire knowledge. Consequently, the distribution of tail entities is largely determined by the knowledge distributions within Wikipedia and Wikidata. In this study, we propose a novel approach to defining tail entities based on their degree information in Wikidata, as opposed to [7] relying on Wikipedia. By doing so, we generate QA datasets with distinct distributions from previous works [7], thus fostering diversity within tail-knowledge QA datasets. Within the context of Wikidata, the degrees of entities reflect their level of engagement with general knowledge. Hence, we leverage this degree information to define tail entities.

The construction of QA datasets typically requires significant human resources, hindering the creation of diverse datasets from various domains that are essential for testing the robustness of current QA models. In this study, our main emphasis lies on the *automatic generation* of long-tail QA datasets. However, we encounter several challenges in this process, such as filtering noisy questions, question granularity, difficulty control, and prompt engineering. These challenges necessitate further research to identify fundamental solutions. We present these challenges through insightful case studies, aiming to stimulate additional research in this area and foster the development of QA models.

Lastly, we assess the performance of pretrained LLMs, specifically GPT3, on our tail entity datasets. Our findings reveal distinct patterns compared to prior work [7], which defines tail entities based on Wikipedia rather than Wikidata. This underscores the importance of utilizing diverse QA sets to accurately gauge the robustness of QA models. Moreover, we investigate strategies to

[*] These authors contributed equally.
[†] Corresponding author: <minjiy@andrew.cmu.edu>
Accepted to Second Workshop on Knowledge Augmented Methods for Natural Language Processing, in conjunction with KDD 2023.

KnowledgeNLP-KDD'23, August 07, 2023, Long Beach, CA
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

enhance the performance of pretrained LLMs by incorporating external resources, such as external documents or knowledge graphs, during inference time on our automatically-generated long-tail QA datasets. We link these experimental results and the challenges encountered during the automatic QA dataset generation process. In summary, our contributions encompass:

- Introduction of novel tail knowledge QA datasets derived from the Wikidata knowledge graph.
- Outline of the automatic construction of QA datasets and associated open research challenges.
- Evaluation of GPT3's performance with/without external resources on our new datasets.

Our code is publicly available[1].

2 RELATED WORK

Fact Learning by LLMs: LLMs [2, 21, 24] have shown state-of-the-art performance across various NLP tasks. LLMs have been shown to memorize facts successfully by learning high-frequency patterns in the training data [7, 9, 20]. Kandpal et al. [7] show that an LLM's ability to answer a question is affected by how many times it has seen relevant documents related to the question in its pre-training data. They show that LLMs struggle to reason accurately over rarer entities in the pre-training data (ROOTS [11], C4 [15], Wikipedia [12], OpenWebText [4]). In this work, instead of using the pre-training corpus, we define tail entities using Wikidata knowledge graphs and construct a long-tail knowledge dataset that can be used to study the open-domain QA performance of LLMs.

Open-domain Question Answering: ODQA is widely used to measure the fact-learning performance of LLMs. However, most of them are composed of common knowledge (i.e., head-entity questions), which prevents deep investigations into LLM's ability to learn facts about uncommon concepts. For instance, TriviaQA [6] is generated from trivia websites, where questions are generally about popular entities or facts. Similarly, NaturalQA [10] is constructed manually using queries issued to the Google search engine. One reason that it is hard to find tail-entity datasets is, most of the time, QA datasets are hand-crafted, requiring a large number of human resources; thus, the types of QA datasets are limited to certain types (e.g., head entities). Here, we focus on how to generate tail-entity datasets while minimizing human resources and analyze why the automatic QA dataset construction is nontrivial.

3 AUTOMATIC GENERATION OF QA DATASETS FOR LONG-TAIL KNOWLEDGE

In this section, we first describe how to generate QA datasets from Wikidata knowledge graphs automatically, then list associated challenges in the process.

3.1 Overview

In knowledge graphs, a triplet [*s1, property, s2*] consists of a subject entity node *s1*, an edge *property*, and an object entity node *s2*. A triplet represents a piece of information about *s1* that can be used to generate a question/answer pair about *s1*. For instance, a triplet [*The Hospital, location, New York City*] represents the information that *The Hospital* is *located* in *New York City*.

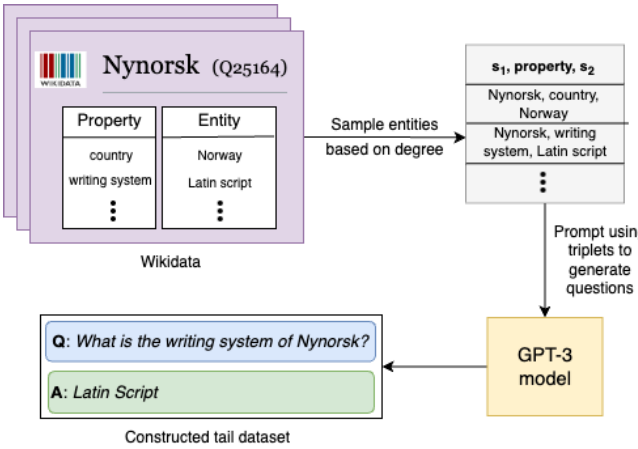

Figure 1: Overview of the automatic QA data construction process for long-tail knowledge: We first sample tail entities that have low degrees and extract the connected triplets from Wikidata knowledge graph; Then we prompt GPT3 with the triplets to generate natural language questions.

We define tail entities based on each entity's node degree (i.e., the number of triplets that have the target entity as a subject node *s1*) in the knowledge graph. We first sample tail entities based on their degree information and extract all triplets that have the tail entities as the subject entity from Wikidata (proper degree bounds of tail entities will be discussed in the following section). Then we generate factoid questions by prompting LLMs with triplets. Specifically, we use the GPT3 model with the following prompt:

```
Generate questions:
obama | born | hawaii => where was obama born?
sky | color | blue => what color is the sky?
X | Y | Z =>
```

Here X, Y, and Z correspond to *s1, property, s2* of triplets extracted from the Wikidata. Figure 1 outlines how to automatically generate QA datasets for tail entities from knowledge graphs.

3.2 Challenges

In this section, we describe open research challenges we faced while constructing QA datasets automatically from knowledge graphs.

3.2.1 Degree bounds for tail entities. There are no strictly-formulated definitions for tail entities that are widely accepted. Degree bounds that instantly bring in differences in model performance are also hard to be decided in advance. As a result, degree bounds for tail entities should be selected arbitrarily. In our experiments, we classify entities with node degrees between 15 and 100 as *coarse*-tail entities and entities with node degrees below 3 as *fine*-tail entities and compare the LLM performance on them. Figure 2 shows the degree distribution of entities in Wikidata.

[1] https://github.com/isunitha98selvan/odqa-tail

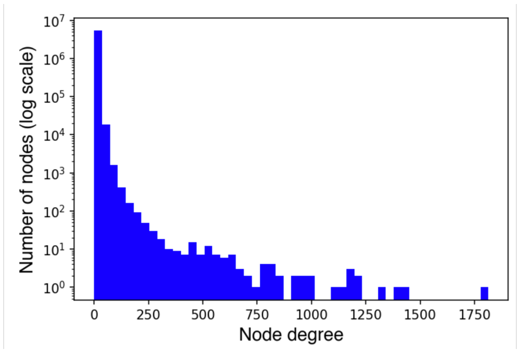

Figure 2: Node degree distribution of all entities in Wikidata.

3.2.2 *Filtering out noisy triplets.*

Ambiguous entities: Multiple entities can have the same surface forms. For instance, *Jesus* can refer to either *1999 Biblical telefilm directed by Roger Young*, *1979 film by Peter Sykes*, or *central figure of Christianity* in Wikidata. While these entities can be distinguished by their unique IDs in knowledge graphs, questions generated from such entities are ambiguous to answer (e.g., *Who was cast in Jesus?*). This introduces complications in evaluating the correctness of a model's answers. Similarly, answer entities can have several correct surface forms. *World War II* can be written as *WW2* , *WWII.* In our experiment, we use correct answers along with their aliases from Wikidata for evaluation. However, aliases provided by Wikidata do not cover all possible surface forms, leading to high false negatives.

Ambiguous properties: In Wikidata, a large number of properties cannot be used to generate sensible questions. For instance, *subclass of*, *instance of*, or *part of* would generate questions that are too vague to answer even for humans. Another example is *family name*, which will generate questions that already contain the answer in them (e.g., *What is the family name of Barack Obama?*). Wikidata also has properties that merely link entities to images or URLs, such as *logo image*, *official website*, and *official blog URL*. Questions generated with these properties are not helpful in evaluating QA model performance, so they need to be filtered out.

As there is no straightforward metric to quantify the appropriateness of each property for question generation, property filtering is difficult to be automated. Property filtering requires human judgment, which can be problematic because it can be subjective as well as difficult to scale. In our experiment, we filter out properties by manually going through all the properties that are initially extracted.

3.2.3 Difficulty control. Questions generated from different properties can have different levels of difficulty. For example, the property *driving side* only has two possible choices, *right* and *left*, for the object (answer) entity. In contrast, the property *child* has approximately 800*k* possible choices for the object entity in Wikidata. The difficulty of selecting the correct answer for these two properties can therefore be very different.

Our goal is to generate long-tail QA sets based on the degree of subject (question) entities in knowledge graphs. In other words, we want the difficulty of questions solely affected by the degree of question entities, not by properties. In Figure 3, we match *coarse*-tail dataset and *fine*-tail dataset to contain the same number of triplets for each property, normalizing the difficulty of QA sets in terms of properties.

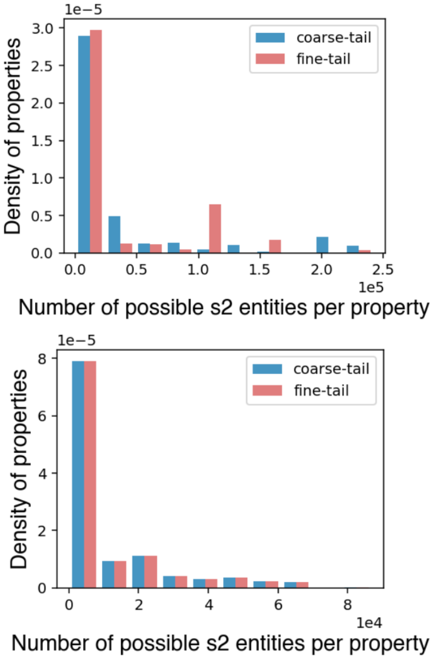

Figure 3: Density of properties per the number of possible *s2* object entities before (Top) and after (Bottom) the difficulty controlling.

3.2.4 LLM prompt for question generation. While the answer entity of a triplet is not part of the generated question, we find that the quality of generated questions improves when the complete triplet is provided in the prompt, instead of the first two elements (i.e., subject entity and property). For instance, given a triplet [*david peel yates, conflict, world war ii*], we get *"What conflict was David Peel Yates involved in?"* from GPT3 when using just the subject entity and property in prompt. On the contrary, when we use all subject, property, and object entities, the generated question becomes *"What conflict did David Peel Yates serve in?"*. By including the answer entity *world war ii* in the prompt, GPT3 understands *conflict* is about a war, not about people, and generates a question with a more proper verb *serve in.* This shows that prompt engineering for generative LLMs is crucial for the quality of generated QA datasets.

3.2.5 Granularity of questions. Given a question, there could be several correct answers with different granularity. Unless the question specifies the granularity of the answer (e.g., *which country* or *which city*), QA datasets and models could easily pick different granularity of answers. For instance, when asked *Where was Lovelyz formed?*, a model could answer *South Korea* while the QA dataset has *Seoul* (the capital of South Korea) as the correct answer and marks the predicted answer wrong. To specify the granularity of the answer in the question, generative LLMs should already know about the question/answer entity, which becomes problematic if these entities are long-tail knowledge. The other solution is preparing all possible granularity as the correct answers, which is also practically infeasible.

4 EVALUATION WITH LLMS AND EXTERNAL RESOURCES

4.1 Experiment Setup

GPT3: We use the *text-davinci-003* version of GPT3 for all our experiments (e.g., question generation, answering questions). The model is accessible via the OpenAI API[2]. Specifically, we make use of the Completions API to prompt the model.

Wikipedia: We use Wikipedia articles to retrieve relevant paragraphs on-the-fly to augment the GPT3 prompt with additional context. To find the relevant paragraphs using Dense Passage Retriever (DPR) [8], we use the same Wikipedia dump as in the original paper [8].

Wikidata: Wikidata knowledge graph consists of 103, 305, 143 entities and 11, 007 properties. We access Wikidata using the Sling tool [17] in a triplet format (*subject*, *property*, *object*).

Tail-entity datasets: We sample triplets from Wikidata to create *Coarse*-tail and *Fine*-tail datasets. Each dataset has 27, 691 triplets and 422 unique properties after the difficulty control (details in Section 3.2.3). One question&answer pair consists of a GPT3-generated question, an answer (i.e., object entity in the original triplet), and associated aliases for the answer.

4.2 LLM prompting for open-domain QA

We study the performance of GPT3 on our tail-entity datasets. We prompt the GPT3 model with few QA pairs as follows:

```
Answer the given question:
where was obama born? => hawaii
what color is the sky? => blue
where was lovelyz formed? =>
```

Table 2 shows GPT3 performance on existing QA datasets (TriviaQA [6], WebQA [1], and NaturalQA [10]) and our newly-generated *Coarse*- and *Fine*-tail QA datasets. GPT3 shows consistently lower performance on our tail datasets than the existing QA datasets, while performing better on *Coarse*-tail set than *Fine*-tail set. This results coincide with [7], showing again that LLMs struggle to learn long-tail knowledge.

We perform manual error analysis on our tail QA dataset. We randomly sample 100 questions that got wrong from *Fine*-tail QA set and categorize their error cases into 6 cases. As shown in Table 1, 45% of errors are from GPT3's completely wrong predictions. 19% of errors are due to different granularity of answers, and 12% of errors are due to questions that are incorrectly generated by LLMs. As we describe in Section 3.2, this result shows the limitations of auto-generated QA datasets and underscores the imperative for further research in this domain.

4.3 LLM prompting with Dense Passage Retrieval

One common way to augment LLMs for long-tail knowledge is retrieving relevant passages from external documents and referring them during inference time [7]. In this section, we check whether GPT3 can see the same improvement in our datasets. We use Dense Passage Retriever (DPR) [8] that has trained on Natural Questions [10] to retrieve the top 100 relevant passages from Wikipedia.

We first evaluate how successfully DPR retrieves a passage that contains the correct answer. In Table 3, we observe that DPR performs consistently worse on our tail datasets than the existing datasets. This shows that DPR, which has pretrained on Wikipedia with head-entity QA datasets, also struggles to retrieve long-tail knowledge.

Next, we use the top-ranked passages retrieved by DPR to augment GPT3. We pass the top-1 retrieved passages to GPT3 as additional context along with the question as follows:

```
Question: Where is Nelson's Pillar located?
Document: Nelson's Pillar was a large granite
column capped by a statue of Horatio Nelson,
built in the centre of what was then Sackville
Street in Dublin, Ireland.
Answer: Dublin, Ireland
```

In Table 4, we observe a decrease in GPT3 accuracy compared to its original accuracy when prompted with DPR retrieved passages. The accuracy of 26.5% for *Coarse*-tail and 22.1% for *Fine*-tail QA sets plummet to 14.3% and 18.2%, respectively. This decline in accuracy can be attributed to the fact that DPR's retrieval often leads to irrelevant passages on long-tail knowledge, as shown in Table 3. Consequently, the presence of these additional contexts confuses GPT3 and adversely affects its performance. These findings highlight the crucial relationship between the performance of LLMs and the retrieval models, indicating that the performance of LLMs is inherently limited by the effectiveness of the retrieval models. Therefore, it is essential for retrieval models to also consider and address the challenges associated with long-tail knowledge.

4.4 LLM prompting with DPR and knowledge graphs

Knowledge graphs (KG) have been widely used to augment LLMs [19, 25]. In this section, we examine how external knowledge graphs can cooperate with another external resource, Wikipedia to improve LLM performance for tail entities. We use Wikidata as our external knowledge graph after removing all triplets used for the QA generation. To avoid additional finetuning, we implement a zero-shot LLM+DPR+KG baseline: we first sample triplets relevant to the question from the knowledge graph then use the sampled triplets to rerank the DPR-retrieved passages; then we pass the top-1 retrieved passages to GPT3 as additional context along with the question. To sample relevant triplets from knowledge graphs, we first find a path from the subject entity to the object entity and concatenate the surface forms of all entities on the path. We then

[2] https://platform.openai.com/docs/api-reference/introduction

Table 1: Error analysis on *Fine*-tail QA set.

Reason	Explanation	Ratio
Incorrect	Wrong answers	45%
Granularity	Answers are too specific or too generic	19%
Incorrect question	Unrelated to input triplets	12%
Exact match	Answers are correct but don't exactly match (e.g., no punctuation, synonyms)	9%
Multiple answers	Both answers and predictions are correct, but questions have multiple answers	3%
Others	e.g., input triplets are not useful/ambiguous	12%

Table 2: GPT3 few-shot performance on open-domain QA datasets. *Results for TriviaQA, WebQA, and NaturalQA are from [2].

Dataset	Accuracy
TriviaQA*	71.2%
WebQA*	41.5%
NaturalQA*	29.9%
Coarse-tail QA	26.5%
Fine-tail QA	22.1%

Table 3: Performance of DPR on retrieving relevant documents from Wikipedia. We check Top-K retrieved passages contain the correct answer. *Results for TriviaQA, WebQA, and NaturalQA are from [8].

Dataset	Top-20	Top-100
TriviaQA*	79.4%	85.0%
WebQA*	73.2%	81.4%
NaturalQA*	78.4%	85.4%
Coarse-tail QA	50.5%	63.3%
Fine-tail QA	54.5%	66.3%

Table 4: Performance of GPT3 prompted with the Top-1 DPR retrieved passage. DPR's low accuracy leads to irrelevant passages being retrieved. Then additional contexts confuse GPT3, leading to a decrease in accuracy compared to its original performance.

	Coarse-tail QA	*Fine*-tail QA
Original	26.5%	22.1%
w/ DPR	14.3%	18.2%

compute its textual similarity with the passages retrieved by DPR using SBERT [16]. We use this similarity score to re-rank the DPR results and observe the changes in the Top-K retrieval accuracy.

As shown in Table 5, DPR retrieval accuracy is improved by up to 6% with the help of knowledge graphs. The improvement in DPR retrieval accuracy leads to the improvement of GPT3's QA performance. Table 5 shows GPT3 also has improved from 22.1% (no external resources) to 30.95% (with DPR and knowledge graphs) on our *Fine*-tail QA datasets. This highlights that joint learning of two external resources could be the key to solving the long-tail knowledge problems.

Table 5: Top-K DPR retrieval accuracy and GPT3 performance on *Fine*-tail QA before/after the retrieved passage re-ranking using knowledge graphs. The third and final columns (*re-rank*) show how Top-K DPR retrieval accuracy and GPT3 performance have changed after the re-ranking.

	DPR	re-rank	*Fine*-tail QA	re-rank
Top-1	23.04%	29.10%	22.10%	30.95%
Top-20	59.56%	65.13%		
Top-50	69.99%	72.44%		
Top-100	75.65%	75.65%		

5 CONCLUSION

Our work highlights the limitations of pre-trained LLMs in handling long-tail knowledge in open-domain Question Answering. To investigate this limitation, we first propose to generate QA datasets specialized for tail entities automatically using degree information from the Wikidata knowledge graph. Our automatic QA generation approach aims to overcome the resource-intensive nature of manual dataset construction, allowing for the creation of diverse long-tail QA datasets. In the process of automatic QA dataset generation, we identify and discuss several open research challenges, such as degree bounds, question granularity, difficulty control, and prompt engineering, which require further investigation for fundamental solutions. We evaluate the performance of GPT3 on our generated long-tail QA datasets. Additionally, we explore the utilization of external resources, such as external documents or knowledge graphs, to improve the performance of LLMs on long-tail knowledge. We hope this work paves the way for further research in the automatic QA dataset generation and the long-tail knowledge problem in open-domain QA tasks.

REFERENCES

[1] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. 1533–1544.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[3] Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179* (2017).

[4] Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus.

[5] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *Comput. Surveys* 55, 12 (2023), 1–38.

[6] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551* (2017).

[7] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2022. Large language models struggle to learn long-tail knowledge. *arXiv preprint arXiv:2211.08411* (2022).

[8] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).

[9] Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are pretrained language models symbolic reasoners over knowledge? *arXiv preprint arXiv:2006.10413* (2020).

[10] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[11] Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, et al. 2022. The bigscience roots corpus: A 1.6 tb composite multilingual dataset. *Advances in Neural Information Processing Systems* 35 (2022), 31809–31826.

[12] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300* (2019).

[13] Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th annual meeting of the Association for Computational Linguistics*. 563–570.

[14] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066* (2019).

[15] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[16] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[17] Michael Ringgaard, Rahul Gupta, and Fernando CN Pereira. 2017. SLING: A framework for frame semantic parsing. *arXiv preprint arXiv:1710.07032* (2017).

[18] Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910* (2020).

[19] Yueqing Sun, Qi Shi, Le Qi, and Yu Zhang. 2021. Jointlk: Joint reasoning with language models and knowledge graphs for commonsense question answering. *arXiv preprint arXiv:2112.02732* (2021).

[20] Michael Tänzer, Sebastian Ruder, and Marek Rei. 2021. Memorisation versus generalisation in pre-trained language models. *arXiv preprint arXiv:2105.00828* (2021).

[21] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[22] Ellen M Voorhees et al. 1999. The trec-8 question answering track report.. In *Trec*, Vol. 99. 77–82.

[23] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[24] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068* (2022).

[25] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860* (2022).